# Практическая работа №22. Задание 1. Программирование линейных алгоритмов

***Цель лабораторной работы:***  научиться составлять каркас простейшей программы в среде Visual Studio. Написать и отладить программу линейного алгоритма.

## 1.1. Структура приложения

Перед началом программирования необходимо создать проект. *Проект* содержит все исходные материалы для приложения, такие как файлы исходного кода, ресурсов, значки, ссылки на внешние файлы, на которые опирается программа, и данные конфигурации, такие как параметры компилятора.

Кроме понятия проект часто используется более глобальное понятие – *решение (solution)*. Решение содержит один или несколько проектов, один из которых может быть указан как стартовый проект. Выполнение решения начинается со стартового проекта.

Таким образом, при создании простейшей C# программы в Visual Studio. Создается папка решения, в которой для каждого проекта создается подпапка проекта, в которой будут создаваться другие подпапки с результатами компиляции приложения.

Проект - это основная единица, с которой работает программист. При создании проекта можно выбрать его тип, а Visual Studio создаст каркас проекта в соответствии с выбранным типом.

В предыдущей лабораторной работе мы попробовали создавать оконные приложения или иначе **Приложения Windows Forms**. Как пример другого типа проекта можно привести проект *консольного* приложения.

По своим "внешним" проявлениям консольные напоминают приложения DOS, запущенные в Windows. Тем не менее, это настоящие Win32-приложения, которые под DOS работать не будут. Для консольных приложений доступен Win32 API, а кроме того, они могут использовать консоль - окно, предоставляемое системой, которое работает в текстовом режиме и в которое можно вводить данные с клавиатуры. Особенность консольных приложений в том, что они работают не в графическом, а в текстовом режиме.

Проект в Visual Studio состоит из файла проекта (файл с расширением *.csproj*), одного или нескольких файлов исходного текста (с расширением *.cs*), файлов с описанием окон формы (с расширением *.designer.cs*), файлов ресурсов (с расширением *.resx*), а также ряда служебных файлах.

В ***файле проекта*** находится информация о модулях, составляющих

данный проект, входящих в него ресурсах, а также параметров построения программы. Файл проекта автоматически создается и изменяется средой Visual Studio и не предназначен для ручного редактирования.

***Файл исходного текста*** – программный модуль предназначен для размещения текстов программ. В этом файле программист размещает текст программы, написанный на языке C#.Модуль имеет следующую структуру:

// Раздел подключенных пространств имен

using System;

// Пространство имен нашего проекта

namespace MyFirstApp

{

// Класс окна

public partial class Form1 : Form

{

// Методы окна

public Form1()

{

InitializeComponent();

}

}

}

В разделе подключения пространств имен (каждая строка которого располагается в начале файла и начинается ключевым словом **using**) описываются используемые пространства имён. Каждое пространство имён включает в себя классы, выполняющие определённую работу, например, классы для работы с сетью располагаются в пространстве System.Net, а для работы с файлами – в System.IO. Большая часть пространств, которые используются в обычных проектах, уже подключена при создании нового проекта, но при необходимости можно дописать дополнительные пространства имён.

Для того чтобы не происходило конфликтов имён классов и переменных, классы нашего проекта также помещаются в отдельное пространство имен. Определяется оно ключевым словом namespace, после которого следует имя пространства (обычно оно совпадает с именем проекта).

Внутри пространства имен помещаются наши классы – в новом проекте это класс окна, который содержит все методы для управления поведением окна. Обратите внимание, что в определении класса присутствует ключевое слово partial, это говорит о том, что в исходном тексте представлена только часть класса, с которой мы работаем непосредственно, а служебные методы для обслуживания окна скрыты в другом модуле (при желании их тоже можно посмотреть, но редактировать вручную не рекомендуется).

Наконец, внутри класса располагаются переменные, методы и другие элементы программы. **Фактически, основная часть программы размещается внутри класса при создании обработчиков событий.**

При компиляции программы Visual Studio создает исполняемые .exe-файлы в каталоге *bin*.

## 1.2. Ввод/вывод данных в программу

Рассмотрим один из способов ввода данных через элементы, размещенные на форме. Для ввода данных чаще всего используют элемент управления TextBox, через обращение к его свойству **Text**. Свойство Text хранит в себе строку введенных символов. Поэтому данные можно считать таким образом:

private void button1\_Click(object sender, EventArgs e)

{

string s = textBox1.Text;

}

Однако со строкой символов трудно производить арифметические операции, поэтому лучше всего при вводе числовых данных перевести строку в целое или вещественное число. Для этого у типов, или int и double существуют методы Parse для преобразования строк в числа. С этими числами можно производить различные арифметические действия. Таким образом, предыдущий пример можно переделать следующим образом:

private void button1\_Click(object sender, EventArgs e)

{

string s = textBox1.Text;

int a = int.Parse(s);

int b = a \* a;

}

Перед выводом числовые данные следует преобразовать назад в строку. Для этого у каждой переменной существует метод ToString, который возвращает в результате строку с символьным представлением значения. Вывод данных можно осуществлять в элементы **TextBox** или **Label**, использую свойство **Text**. Например:

private void button1\_Click(object sender, EventArgs e)

{

string s = textBox1.Text;

int a = int.Parse(s);

int b = a \* a;

label1.Text = b.ToString();

}

## 1.3. Пример написания программы

Задание: составить программу вычисления для заданных значений x, y, z арифметического выражения

.

Панель диалога программы организовать в виде, представленном на рис:

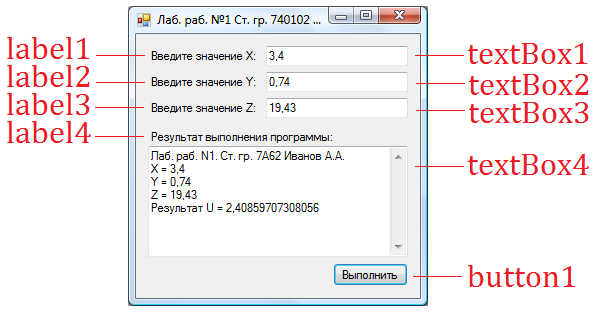


Рис 1.1. Внешний вид программы.

Для вывода результатов работы программы в программе используется текстовое окно, которое представлено обычным элементом управления. После установки свойства **Multiline** в **True** появляется возможность растягивать элемент управления не только по горизонтали, но и по вертикали. А после установки свойства **ScrollBars** в значение **Both** в окне появится вертикальная, а при необходимости и горизонтальная полосы прокрутки.

Информация, которая отображается построчно в окне, находится в массиве строк **Lines**, каждая строка которого имеет тип **string**. Однако нельзя напрямую обратиться к этому свойству для добавления новых строк, поскольку размер массивов в C# определяется в момент их инициализации. Для добавления нового элемента используется свойство Text, к текущему содержимому которого можно добавить новую строку:

textBox4.Text += Environment.NewLine + "Привет";

В этом примере к текущему содержимому окна добавляется символ перевода курсора на новую строку (который может отличаться в разных операционных системах и потому представлен свойством класса Environment) и сама новая строка. Если добавляется числовое значение, то его предварительно нужно привести в символьный вид методом ToString().

Работа с программой происходит следующим образом. Нажмите (щелкните мышью) кнопку “Выполнить”. В окне textBox4 появляется результат. Измените исходные значения x, y, z в окнах textBox1 – textBox3 и снова нажмите кнопку ”Выполнить” - появится новые результаты.

Полный текст программы имеет следующий вид:

using System;

using System.Windows.Forms;

namespace MyFirstApp

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

textBox1.Text = "3,4"; // Начальное значение X

textBox2.Text = "0,74"; // Начальное значение Y

textBox3.Text = "19,43";// Начальное значение Z

// Вывод строки в многострочный редактор

textBox4.Text = "Лаб. раб. N1. Ст. гр. 7А62 Иванов А.А.";

}

private void button1\_Click(object sender, EventArgs e)

{

// Считывание значения X

double x = double.Parse(textBox1.Text);

// Вывод значения X в окно

textBox4.Text += Environment.NewLine +

"X = " + x.ToString();

// Считывание значения Y

double y = double.Parse(textBox2.Text);

// Вывод значения Y в окно

textBox4.Text += Environment.NewLine +

"Y = " + y.ToString();

// Считывание значения Z

double z = double.Parse(textBox3.Text);

// Вывод значения Z в окно

textBox4.Text += Environment.NewLine +

"Z = " + z.ToString();

// Вычисляем арифметическое выражение

double a = Math.Tan(x + y) \* Math.Tan(x + y);

double b = Math.Exp(y - z);

double c = Math.Sqrt(Math.Cos(x \* x) + Math.Sin(z \* z));

double u = a -b \* c;

// Выводим результат в окно

textBox4.Text += Environment.NewLine +

"Результат U = " + u.ToString();

}

}

}

## 1.4. Выполнение индивидуального задания

Ниже приведено 15 вариантов задач. По указанию преподавателя выберите свое индивидуальное задание. Уточните условие задания, количество, наименование, типы исходных данных. В соответствии с этим установите необходимое количество окон TextBox, тексты заголовков на форме, размеры шрифтов, а также типы переменных и функции преобразования при вводе и выводе результатов.

Прочтите в приложении описание меню Файл, Правка, Отладка, и описание элемента управления TextBox. С помощью инспектора объектов измените цвет формы, шрифт выводимых символов.

## Индивидуальные задания

1. 

При x=14.26, y=-1.22, z=3.5× t=0.564849.

2. 

При x=-4.5, y=0.75×,z=0.845 u=-55.6848.

3. 

При x=3.74, y=-0.825, z=0.16, *v*=1.0553.

4. 

При x=0.4 ,y=-0.875, z=-0.475 *w*=1.9873.

5. 

При x=-15.246, y=4.642, z=20.001 =-182.036.

6. 

При x=16.55, y=-2.75, z=0.15 =-38.902.

7. 

При x=0.1722, y=6.33, z=3.25 =-172.025.

8. 

При x=-2.235, y=2.23, z=15.221 =39.374.

9. 

При x=1.825, y=18.225, z=-3.298 =1.2131.

10. 

При x=3.981, y=-1.625, z=0.512 *a*=1.26185.

11. 

При x=6.251, y=0.827, z=25.001 *b*=0.7121.

12.

При x=3.251, y=0.325, z=0.466 *c*=4.025.

13. .

При x=17.421, y=10.365, z=0.828 *f*=0.33056.

14. .

При x=12.3, y=15.4, z=0.252 *g*=82.8257.

15. .

При x=2.444, y=0.869, z=-0.13 -0.49871.

# Практическая работа №22. Задание 2. Программирование разветвляющихся алгоритмов

***Цель лабораторной работы****:* научиться пользоваться простейшими компонентами организации переключений (RadioButton). Написать и отладить программу разветвляющегося алгоритма.

## 2.1. Кнопки-переключатели RadioButton

При создании программ в Visual Studio для организации разветвлений часто используются компоненты в виде кнопок-переключателей. Состояние такой кнопки (включено - выключено) визуально отражается на форме. Если пользователь выбирает один из вариантов переключателя в группе, все остальные автоматически отключаются.

Группу составляют все элементы управления **RadioButton** в заданном контейнере, таком как **Form**. Чтобы создать на одной форме несколько групп, поместите каждую группу в собственный контейнер, такой как элемент управления **GroupBox** или **Panel**. На форме (рис.3.1) представлены кнопки-переключатели **RadioButton** в контейнере **GroupBox**.

В программу передается номер включенной кнопки (0,1,2,..), который анализируется с помощью оператора **switch**.

## 2.2. Пример написания программы

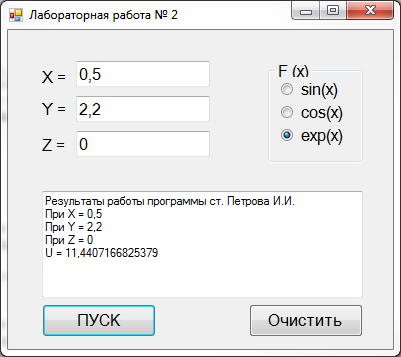
Задание: ввести три числа - x,y,z. Вычислить



В качестве f(x) использовать по выбору: sin(x), cos(x), ex.

## 2.2.1. Создание формы

Создайте форму, в соответствии с рис. 2.1.



*Рис 2.1. Окно лабораторной работы*

Выберите в панели элементов из контейнеров **GroupBox** и поместите его в нужное место формы. На форме появится окаймленный линией чистый прямоугольник с заголовком **GroupBox1**. Замените заголовок (**Text**) на F(x). Далее, как показано на рисунке, разместите в данном контейнере три радиокнопки (**RadioButton**). Для первой из них установите свойство **Checked** в значение **True**.

Далее разместите на форме элементы **Label**, **TextBox** и **Button**. Поле для вывода результатов также является элементом **TextBox** с установленным в **True** свойством **Multiline**.

## 2.2.2. Создание обработчиков событий FormСreate и Botton1Сlick

Обработчики событий создаются аналогично тому, как и в предыдущих лабораторных работах. Текст обработчика события нажатия на кнопку ПУСК приведен ниже.

private void button1\_Click(object sender, EventArgs e)

{ // Получение исходных данных из TextBox

double x = Convert.ToDouble(textBox1.Text);

double y = Convert.ToDouble(textBox2.Text);

double z = Convert.ToDouble(textBox3.Text);

// Ввод исходных данных в окно результатов

textBox4.Text = "Результаты работы программы ст. Петрова И.И. " + Environment.NewLine;

textBox4.Text += "При X = " + textBox1.Text + Environment.NewLine;

textBox4.Text += "При Y = " + textBox2.Text + Environment.NewLine;

textBox4.Text += "При Z = " + textBox3.Text + Environment.NewLine;

// Определение номера выбранной функции

int n = 0;

if (radioButton2.Checked) n = 1;

else if (radioButton3.Checked) n = 2;

// Вычисление U

double u;

switch (n)

{

case 0:

if ((z - x) == 0) u = y \* Math.Sin(x) \* Math.Sin(x) + z;

else if ((z - x) < 0) u = y\*Math.Exp(Math.Sin(x)) - z;

else u = y\*Math.Sin(Math.Sin(x)) + z;

textBox4.Text += "U = " + Convert.ToString(u) + Environment.NewLine;

break;

case 1:

if ((z - x) == 0) u = y \* Math.Cos(x) \* Math.Cos(x) + z;

else if ((z - x) < 0) u = y\*Math.Exp(Math.Cos(x)) - z;

else u = y\*Math.Sin(Math.Cos(x)) + z;

textBox4.Text += "U = " + Convert.ToString(u) + Environment.NewLine;

break;

case 2:

if ((z - x) == 0) u = y \* Math.Exp(x) \* Math.Exp(x) + z;

else if ((z - x) < 0) u = y\*Math.Exp(Math.Exp(x)) - z;

else u = y\*Math.Sin(Math.Exp(x)) + z;

textBox4.Text += "U = " + Convert.ToString(u) + Environment.NewLine;

break;

default:

textBox4.Text += "Решение не найдено" + Environment.NewLine;

break;

}

}

Запустите программу и убедитесь в том, что все ветви алгоритма выполняются правильно.

## 2.3. Выполнение индивидуального задания

По указанию преподавателя выберите индивидуальное задание из нижеприведенного списка. В качестве f(x) использовать по выбору: sh(x), x2, ex. Отредактируйте вид формы и текст программы, в соответствии с полученным заданием.

|  |  |  |  |
| --- | --- | --- | --- |
| 1. |  | 2. |  |
| 3. |  | 4. |  |
| 5. |  | 6. |  |
| 7. |  | 8. |  |
| 9. |  | 10. |  |
| 11. |  | 12. |  |
| 13. |  | 14. |  |

15. 

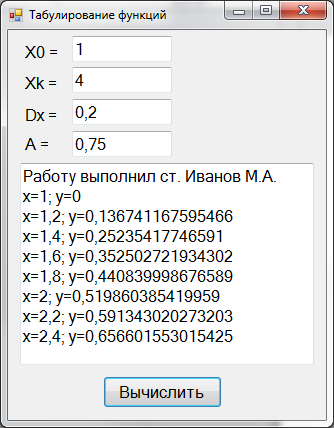
# Практическая работа №22. Задание 3. Программирование циклических алгоритмов

***Цель лабораторной работы:*** изучить простейшие средства отладки программ в среде Visual Studio. Составить и отладить программу циклического алгоритма.

## 3.1. Порядок выполнения задания

Задание: Вычислим и выведем на экран таблицу значений функции y=a•lnx при x, изменяющемся от x0 до xk с шагом dx, a - константа.

Панель диалога представлена на рис 3.1.



*Рис. 3.1. Окно программы для табулирования функции.*

Текст обработчика нажатия кнопки **Вычислить** приведен ниже.

private void button1\_Click(object sender, EventArgs e)

{

// Считывание начальных данных

double x0 = Convert.ToDouble(textBox1.Text);

double xk = Convert.ToDouble(textBox2.Text);

double dx = Convert.ToDouble(textBox3.Text);

double a = Convert.ToDouble(textBox4.Text);

textBox5.Text = "Работу выполнил ст. Иванов М.А." + Environment.NewLine;

// Цикл для табулирования функции

double x = x0;

while (x<=(xk+dx/2))

{

double y = a\*Math.Log(x);

textBox5.Text += "x=" + Convert.ToString(x) +

"; y=" + Convert.ToString(y) + Environment.NewLine;

x = x + dx;

}

После отладки программы составьте тест (X0=2, Xk=4, Dx=0,2), установите точку останова на оператор перед циклом и запустите программу в отладочном режиме (F5). После попадания на точку остановки, нажимая клавишу F10, выполните пошагово программу и проследите, как меняются все переменные в процессе выполнения.

## 3.2. Выполнение индивидуального задания

По указанию преподавателя выберите нужный вариант задачи из нижеприведенного списка. Откорректируйте панель диалога и текст программы.

## Индивидуальные задания

Cоставить программу для табулирования функции y(x), вывести на экран значения x и y(x)

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# ПРактическая работа № 22. Задание 4. Программирование с использованием строк

***Цель лабораторной работы***: изучить правила работы с компонентом ListBox. Написать программу для работы со строками.

## 4.1. Компонент ListBox

Компонент **ListBox** представляет собой список, элементы которого выбираются при помощи клавиатуры или мыши. Список элементов задается свойством **Items**. Items – это элемент, который имеет свои свойства и свои методы. Методы **Add**, **RemoveAt** и **Insert** используются для добавления, удаления и вставки элементов.

Объект **Items** хранит объекты, находящиеся в списке. Объект может быть любым классом – данные класса преобразуются для отображения в строковое представление методом ToString. В нашем случае в качестве объекта будут выступать строки. Однако, поскольку объект Items хранит объекты, приведённые к типу object, перед использованием необходимо привести их обратно к изначальному типу, в нашем случае string:

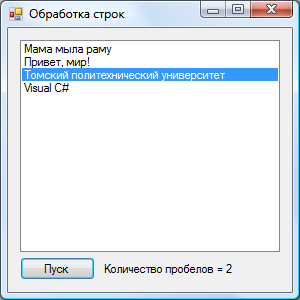
string a = (string)listBox1.Items[0];

Для определения номера выделенного элемента используется свойство **SelectedIndex**.

## 4.2. Порядок выполнения индивидуального задания

Задание: Написать программу подсчета числа слов в произвольной строке. В качестве разделителя может быть любое число пробелов. Для ввода строк использовать **ListBox**. Строки вводятся на этапе проектирования формы, используя окно свойств. Вывод результата организовать в метку **Label**.

Панель диалога будет иметь вид:



*Рис. 4.1. Окно программы обработки строк*

Текст обработчика нажатия кнопки «Пуск» приведен ниже.

private void button1\_Click(object sender, EventArgs e)

{

// Получаем номер выделенной строки

int index = listBox1.SelectedIndex;

// Считываем строку в перменную str

string str = (string)listBox1.Items[index];

// Узнаем количество символов в строке

int len = str.Length;

// Считаем, что количество пробелов равно 0

int count = 0;

// Устанавливаем счетчик символов в 0

int i = 0;

//Организуем цикл перебора всех символов в строке

while (i < len - 1)

{

// Если нашли пробел, то увеличиваем

// счетчик пробелов на 1

if (str[i] == ' ')

count++;

i++;

}

label1.Text = "Количество пробелов = " +

count.ToString();

}

## 4.3. Индивидуальные задания

Во всех заданиях исходные данные вводить с помощью **ListBox**. Строки вводятся на этапе проектирования формы, используя окно свойств. Вывод результата организовать в метку **Label**.

1. Дана строка, состоящая из групп нулей и единиц. Посчитать количество нулей и единиц.

2. Посчитать в строке количество слов.

3. Найти количество знаков препинания в исходной строке.

4. Дана строка символов. Вывести на экран цифры, содержащиеся в строке.

5. Дана строка символов, состоящая из произвольных десятичных цифр, разделенных пробелами. Вывести количество четных чисел в этой строке.

7. Дана строка символов. Вывести на экран количество строчных русских букв, входящих в эту строку.

8. Дана строка символов. Вывести на экран только строчные русские буквы, входящие в эту строку.

9. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. В каждом слове заменить первую букву на прописную.

10. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. Удалить первую букву в каждом слове.

11. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. Поменять местами *i*- и *j*-ю буквы. Для ввода *i* и *j* на форме добавить свои поля ввода.

12. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. Поменять местами первую и последнюю буквы каждого слова.

13. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. Заменить все буквы латинского алфавита на знак ‘+’.

14. Дана строка символов, содержащая некоторый текст на русском языке. Заменить все большие буквы буквы ‘А’ на символ ‘\*’.

15. Дана строка символов, содержащая некоторый текст. Разработать программу, которая определяет, является ли данный текст палиндромом, т.е. читается ли он слева направо так же, как и справа налево ( например, «А роза упала на лапу Азора»).

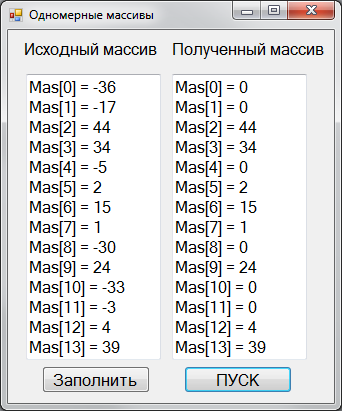
16. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. Сформировать новую строку, состоящую из чисел длин слов в исходной строке.

# ПРактическая работа № 22. Задание 5. Программирование с использованием одномерных массивов

***Цель лабораторной работы****:* Изучить способы получения случайных чисел. Написать программу для работы с одномерными массивами.

## 5.1. Порядок выполнения индивидуального задания

Создайте форму с элементами управления как приведено на рис. 5.1. Опишите одномерный массив. Создайте обработчики события для кнопок (код приведен ниже). Данная программа заменяет все отрицательные числа нулями. Протестируйте правильность выполнения программы. Модифицируйте программу в соответствии с индивидуальным заданием.



*Рис. 5.1. Окно программы для работы с одномерными массивами*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace WindowsFormsApplication1

{

public partial class Form1 : Form

{

int[] Mas = new int[15];

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

Random rand = new Random();

textBox1.Text = "";

for (int i = 0; i < 15; i++)

{

Mas[i] = rand.Next(-50, 50);

textBox1.Text += "Mas[" + Convert.ToString(i) +"] = "

+ Convert.ToString(Mas[i]) + Environment.NewLine;

}

}

private void button2\_Click(object sender, EventArgs e)

{

textBox2.Text = "";

for (int i = 0; i < 15; i++)

{

if (Mas[i] < 0) Mas[i] = 0;

textBox2.Text += "Mas[" + Convert.ToString(i) + "] = "

+ Convert.ToString(Mas[i]) + Environment.NewLine;

}

}

}

}

## 5.2. Варианты заданий

1. В массиве из 20 целых чисел найти наибольший элемент и поменять его местами с первым элементом.
2. В массиве из 10 целых чисел найти наименьший элемент и поменять его местами с последним элементом.
3. В массиве из 15 вещественных чисел найти наибольший элемент и поменять его местами с последним элементом.
4. В массиве из 25 вещественных чисел найти наименьший элемент и поменять его местами с первым элементом.
5. Дан массив X, содержащий 27 элементов. Вычислить и вывести элементы нового массива Y, где yi =6.85xi2-1.52. Если yi<0, то вычислить и вывести a=xi3-0.62 и продолжить вычисления; если yi≥0, то вычислить и вывести b=1/xi2 и продолжить вычисления.
6. Дан массив X, содержащий 16 элементов. Вычислить и вывести значения di, где  и значения di>0.1.
7. Дан массив Y, содержащий 25 элементов. Записать в массив R и вывести значения элементов, вычисляемые по формуле , i=1,2,…,25.
8. Дан массив F, содержащий 18 элементов. Вычислить и вывести элементы нового массива pi=0.13fi3-2.5fi+8. Вывести отрицательные элементы массива P.
9. Вычислить и вывести элементы массива Z, где zi=i2+1, если i – нечетное, и zi=2i-1, если i – четное. Сформировать и вывести массив D: di=2.5zi, если zi<2.5 и di=zi/2.5, если zi≥2.5.
10. Заданы массивы D и E. Вычислить и вывести значения fi=(2di+sin ei)/di, где i=1,2,…,16; вывести 1<fi<3.
11. В массиве R, содержащем 25 элементов, заменить значения отрицательных элементов квадратами значений, значения положительных увеличить на 7, а нулевые значения оставить без изменения. Вывести массив R.
12. Дан массив A целых чисел, содержащий 30 элементов. Вычислить и вывести сумму тех элементов, которые кратны 5.
13. Дан массив A целых чисел, содержащий 30 элементов. Вычислить и вывести сумму тех элементов, которые нечетны и отрицательны.
14. Дан массив A целых чисел, содержащий 30 элементов. Вычислить и вывести сумму тех элементов, которые удовлетворяют условию |ai|<i2.
15. Дан массив A целых чисел, содержащий 30 элементов. Вычислить и вывести количество и сумму тех элементов, которые делятся на 5 и не делятся на 7.
16. Дан массив A вещественных чисел, содержащий 25 элементов. Вычислить и вывести число отрицательных элементов и число членов, принадлежащих отрезку[1,2].
17. Дан массив C, содержащий 23 элемента. Вычислить и вывести среднее арифметическое всех значений ci>3.5.
18. Дан массив Z целых чисел, содержащий 35 элементов. Вычислить и вывести R=S+P, где S – сумма четных элементов, меньших 3, P – произведение нечетных элементов, больших 1.
19. Дан массив Q натуральных чисел, содержащий 20 элементов. Найти и вывести те элементы, которые при делении на 7 дают остаток 1,2 или 5.
20. Дан массив Q натуральных чисел, содержащий 20 элементов. Найти и вывести те элементы, которые обладают тем свойством, что корни уравнения qi2+3qi-5=0 действительны и положительны.
21. Дан массив, содержащий 10 элементов. Вычислить произведение элементов, стоящих после первого отрицательного элемента. Вывести исходный массив и результат вычислений.
22. Дан массив, содержащий 14 элементов. Вычислить сумму элементов, стоящих до первого отрицательного элемента. Вывести исходный массив и результат вычислений.
23. Дан массив содержащий 12 элементов. Все четные элементы сложить, вывести массив и результат.
24. Дан массив, содержащий 15 элементов. Все положительные элементы возвести в квадрат, а отрицательные умножить на 2. Вывести исходный и полученный массив.
25. Дан массив, содержащий 14 элементов. Все отрицательные элементы заменить на 3. Вывести исходный и полученный массив.

# Практическая РАБОТА 22. Задание 6. Программирование с использованием МНОГОМЕРНЫХ массивов

***Цель лабораторной работы****:* изучить свойства компонента **dataGridView**. Написать программу с использованием двухмерных массивов.

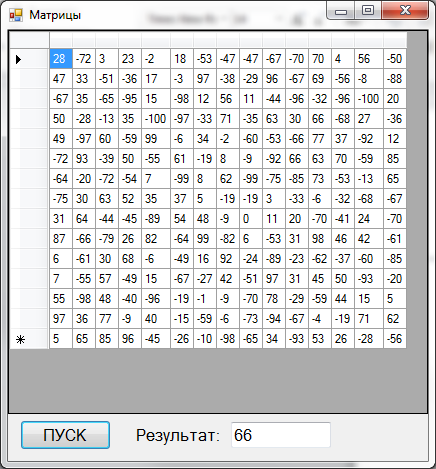
## 6.1. Элемент управления DataGridView

При работе с двухмерными массивами ввод и вывод информации на экран удобно организовывать в виде таблиц. Элемент управления **DataGridView** может быть использован для отображения информации в виде двумерной таблицы. Для обращения к ячейке в этом элементе необходимо указать номер строки и номер столбца. Например: **dataGridView1.Rows[2].Cells[7].Value = "\*";**  данный код позволяте записать во вторую строку в 7 ячейку знак звездочка.

## 6.2. Порядок выполнения задания

Задание: Создать программу для определения целочисленной матрицы 15 на 15. Разработать обработчик для поиска минимального элемента на дополнительной диаганали матрицы. Результат, после нажатия кнопки типа **Button**, вывести в **textBox**.

Окно программы приведено на рис. 6.1.



*Рис. 6.1. Окно программы для работы с двухмерным массивом*

Текст обработчика события нажатия на кнопку приведен ниже.

private void button1\_Click(object sender, EventArgs e)

{

dataGridView1.RowCount = 15; //Указываем количество строк

dataGridView1.ColumnCount = 15; //Указываем количество столбцов

int[,] a = new int[15,15]; //Инициализируем массив

int i,j;

//Заполняем матрицу случайными числами

Random rand = new Random();

for (i=0; i<15; i++)

for (j=0; j<15; j++)

a[i,j] = rand.Next(-100,100);

//Выводим матрицу в dataGridView1

for (i=0; i<15; i++)

for (j=0; j<15; j++)

dataGridView1.Rows[i].Cells[j].Value = Convert.ToString(a[i,j]);

//производим поиск максимального элемента на дополнительной диагонали

int m = int.MinValue;

for (i = 0; i < 15; i++)

if (a[i, 14 - i] > m) m = a[i, 14 - i];

// выводим результат

textBox1.Text = Convert.ToString(m);

}

## 6.3. Индивидуальные задания

1. Дана матрица A(3,4). Найти наименьший элемент в каждой строке матрицы. Вывести исходную матрицу и результаты вычислений.
2. Дана матрица A(3,3). Вычислить сумму второй строки и произведение первого столбца. Вывести исходную матрицу и результаты вычислений.
3. Дана матрица A(4,4). Найти наибольший элемент в главной диагонали. Вывести матрицу и наибольший элемент.
4. Дана матрица A(3,4). Найти сумму элементов главной диагонали и эту сумму поставить на место последнего элемента. Вывести исходную и полученную матрицу.
5. Дана матрица A(4,3). Вычислить наибольший элемент матрицы. Вывести исходную матрицу и наибольший элемент.
6. Дана матрица A(4,3).Найти количество положительных элементов.
7. Дана матрица A(3,4).Найти количество отрицательных элементов.
8. Даны матрицы X(15,15) и Y(15,15). Вычислить и вывести элементы новой матрицы zij=12xij-0.85yij2.
9. Даны матрицы A(6,6),B(6,6) и C(6,6). Получить матрицу D(6,6), элементы которой вычисляются по формуле dij=max{aij,(bij+cij)}. Матрицу D(6,6) вывести.
10. Вычислить сумму S элементов главной диагонали матрицы B(10,10). Если S>10, то исходную матрицу преобразовать по формуле bij=bij+13.5; если S≤10, то bij=bij2-1.5. Вывести сумму S и преобразованную матрицу.
11. Дана матрица F(15,15). Вывести номер и среднее арифметическое элементов строки, начинающейся с 1. Если такой строки нет, то вывести сообщение “строки нет”.
12. Дана матрица F(7,7). Найти наименьший элемент в каждом столбце. Вывести матрицу и найденные элементы.
13. Найти наибольший элемент главной диагонали матрицы A(15,15) и вывести всю строку, в которой он находится.
14. Найти наибольшие элементы каждой строки матрицы Z(16,16) и поместить их на главную диагональ. Вывести полученную матрицу.
15. Вычислить суммы элементов матрицы Y(12,12) по столбцам и вывести их.
16. Найти наибольший элемент матрицы A(10,10) и записать нули в ту строку и столбец, где он находится. Вывести наибольший элемент, исходную и полученную матрицу.
17. Дана матрица R(9,9). Найти наименьший элемент в каждой строке и записать его на место первого элемента строки. Вывести исходную и полученную матрицы.
18. Определить количество положительных элементов каждой строки матрицы A(10,20) и запомнить их в одномерном массиве N. Массив N вывести.
19. Вычислить количество H положительных элементов последнего столбца матрицы X(5,5). Если H<3, то вывести все положительные элементы матрицы, если H≥3, то вывести сумму элементов главной диагонали матрицы.
20. Вычислить и вывести сумму элементов матрицы A(12,12), расположенных над главной диагональю матрицы.